أسس البرمجة بلغة دلفي

(شرح مبسط للراغبين في فهم مسار لغات البرمجة بصفة عامة و لغة دلفي على وجه الخصوص)

بسم الله الرحمن الرحيم

تمهيد:

سنقوم في هذا الكتيب إن شاء الله باقتحام عالم البرمجة الحاسوبية, متخذين من لغة دلفي مطية لبلوغ الغاية التي يصبوا إليها كل مبرمج مبتدئ أو أي شخص ينوي البداية في هذا العالم الرحب.
عالم البرمجة بكل تأكيد يرحب بكل ضيوفه و لا يميز بين عالم رياضيات تلقى تعليمه بأسلوب أكاديمي أو هاو لهذا المجال.

يجدر بنا الذكر أن لغات البرمجة تتطلب تفرغ ذهني , و أأكد على هذه العبارة, كما أعتقد أن أيّ إغفال أو ابتعاد عن العالم السيبراني قد يكلف المبرمج إضاعة الكثير من الوقت لمراجعة ما تعلمه سابقا.لهذا ينبغي على كل من أراد الخوض في هذا المجال أن يحرص كل الحرص على تخصيص الكثير من الوقت لتعلم البرمجة , أيّ لغة برمجة, سواء أ كانت من عهد الستينات أو وليدة اليوم.كذلك ينبغي أن لا يشعر المرء بالإحباط مستسلما لليأس في حالة ما أمضى الكثير من الوقت دون الوصول إلى نتيجة معينة, فقد لاحظت أن الكثير من المبرمجين يستسلمون بمجرد أن يواجههم عائق بسيط.فقد يقول مبرمج اليوم كيف لي أن أتابع البرمجة و أنا لم أستطيع حل مشكلة بسيطة كهذه أمام الكم الهائل من الأدوات و التسهيلات التي أضحت متوفرة , بينما أسلافنا في عصر الكوبول استطاعوا عمل الكثير و إن قلّت الإمكانيات.

جرت العادة عند إختيار عمل ما أن يطرح السؤال التالي*: لماذا هذا العمل بالذات؟*
نحن لن نجيب عن هذا السؤال لأنه متعلق بالقارئ بصفة خاصة و حديثنا عن البرمجة اليوم يبدو بشكل عام و إن كنا قد إخترنا الدلفي لكثير من الصفات المميزة فيها.لكن و من أجل الوصول إلى جواب تقريبي للسؤال المطروح , لا بد لنا من تصنيف المبرمجين كي يعرف كل شخص الإجابة التي تخصه.
يمكننا أن نصنف المبرمجين إلى صنفين :

1- *أكادميين:* و هم الذين تلقوا تعليمهم بأسلوب أكاديمي في الجامعات و معاهد الكمبيوتر و الإعلام الآلي و الهندسة.و تتميز هذه الفئة بالدقة في البرمجة نظرا لتكون الفكر البرمجي لدى الشخص, كون أنه تلقى تعليمه وفقا لمنهج مسطر من قبل خبراء.فلا يمكن على سبيل المثال أن تجد مبرمجا بلغة دلفي لم يسبق له العمل على باسكال و حتى في يومنا هذا, أي بعد أن أصبحت هذه اللغة بمترجمها الشهير من خبر كان, ذلك لأنه تم دمجها في لغة دلفي. مع كل هذا لا يمكن الجزم بأن هذه الفئة تمتلك زمام الأمور فكما يعلم الكثير من الإخوة أن جامعاتنا العربية تلتزم بالجانب النظري أكثر من الجانب التطبيقي. فلا تستغرب إذا ما وجدت مبرمجا هاو يشرح بعض الأمور في لغة دلفي للأكاديمي و إن كان هذا الأخير يحمل قاعدة منوطة بالمعرفة الشاملة لخفايا البرمجة.

2- *هواة:* يمضون الكثير من الوقت أمام شاشة الكمبيوتر مداعبين لوحة المفاتيح و الفأرة.تتميز هذه الفئة بتلقيها الفكر البرمجي من خلال المطالعة الشخصية أي بشكل عصامي. إن الإرادة القوية و حب اكتشاف المجهول هو الأمر اللافت للانتباه لدى هذه الفئة, فعلى الرغم من أنها لم تتلقى تعليمها وفقا لمناهج مسطرة إلا أنها تستطيع بلوغ أهدافها بسهولة خصوصا و أن الشبكة المعلوماتية أصبحت كالمصباح السحري .أكتب أي عبارة تريد البحث عنها في *google* أو باقي محركات البحث و ستظهر لك العشرات من النتائج.عموما يمكن القول أن عدم الالتزام بأي واجبات برمجية هو أحد أسرار هذه الفئة ثم إن مفهوم تحويل كل حتمية أو عائق إلى محفز هو ثاني أحد أهم أسرار الهواة.

كذلك يمكننا التمييز بين المبرمجين وفقا لحاجاتهم البرمجية فالذي يعمل في مؤسسة للبرمجيات ضمن فريق عمل متكامل بكل تأكيد ليس كالشخص الذي يعمل بمفرده. فإذا كان الأول يعمل على تطوير برامج موجهة للسوق تتطلب على الأقل أن لا تتلقى استياء الزبون. فإن الثاني تقتصر أعماله على برامج صغيرة تتداول على مواقع الإنترنت ليستفاد منها ,سواءا أكانت مجانية أو تتطلب الدفع المسبق للحصول عليها.أقول أن الذي يعمل بمفرده تجده ممزقا بين الكثير من المهام, فإذا كان يطور برنامج قاعدة بيانات فلن يجد زميلا لأداء المهام عنه, لأنه يعمل بمفرده.لذا عادة ما نجد أن الهواة يتقنون برمجة تطبيقات ويندوز و برمجة الإنترنت و العمل على قواعد البيانات فضلا عن العمل على برامج الرسم و الفلاش بلاير و باقي البرامج التي عادة ما تدمج في أي تطبيق. لكن الشيء الملاحظ أن الهواة منهم من يهدف إلى

الانضمام إلى فريق عمل و إن كان لا يملك المؤهل (شهادة التخصص في مجال البرمجة) و منهم من يكتفي بخدمة أغراض خاصة أو دفاعا عن مبادئ عامة. لا يفهم مما سبق أن الذين يعملون ضمن فريق عمل لا يدافعون عن مبادئ خاصة أو عامة أو أنهم لا يتقنون باقي التقنيات و إنما يعكس ما سبق ذكره الحتمية المفروضة على الهاوي للبرمجة.

من هنا يستطيع المرء أن يجيب عن السؤال الكلاسكي :

لماذا البرمجة بالذات ؟ لكسب لقمة العيش, للدفاع عن حرمة الإسلام , لنشر الثقافات المحلية , للرقي بالعالم العربي في المجال التقني. لمسايرة عصر تدفق المعلومات.لمساعدة الوالد في المتجر أو الصديق في ...

لغات البرمجة في لمحة:

إن الكمبيوتر مجرد ألة بسيطة تؤدي مجموعة من المهام المختلفة, كالعد على سبيل المثال.إن مهام الكمبيوتر لا تتم إلا وفق سلسلة من الأوامر المترابطة و التي تسمى تطبيقا. يتم تصميم التطبيق من قبل شخص يدعى المبرمج. يعتمد كل مبرمج على لغة برمجة معينة تستقل بتعليمات , تنسيق, كلمات محجوزة و تركيب معين يميزها عن باقي لغات البرمجة.تختلف البرمجة من لغة إلى أخرى من حيث التعقيد , فقد تبدو لغات اليوم أقرب إلى لغة الإنسان, بينما في الماضي كانت أقرب للغة الآلة.

1- *لغة الآلة:* هي اللغة الوحيدة التي يفهمها الكمبيوتر بشكل مباشر, فهي تعتمد بالأساس على تعليمات ثنائية, لهذا تسمى باللغة الثنائية.حيث أن أيّ أمر يوجه إلى الكمبيوتر ينبغي أن يتألف من مجموعة من الأصفار و الآحاد (0,1)حيث يعني الصفر off و يعني الواحد on .مثلا لو أراد الشخص توجيه أمر للكمبيوتر بطباعة حرف A على سبيل المثال فلا بد أن ينشأ الأمر على الشكل التالي: 10101001 . يتم توزيع الأرقام السابقة وفق ترتيب معين يصعب إن لم نقل يستحيل على الإنسان حفظه .تخيل معي أنك تود أن تكتب فقرة صغيرة بلغة الآلة.إن هذا يتطلب معرفة جيدة للتركيب الداخلي للكمبيوتر و كذا العناوين الرقمية لمواقع التخزين. و مع ذلك سنشهد عشرات الأخطاء .
إن كل كمبيوتر ينفرد بلغة ألة خاصة به مما يعني أن البرنامج الذي صممته للتعامل مع هذا الجهاز قد لا يعمل على الجهاز الأخر و هذا ما يزيد الأمر تعقيدا.أمام هذا الوضع المعقد تم إبتكار نظام عد آخر لتمثيل النظام الثنائي مثل النظام الست عشري *(HEXADECIMAL)* الذي يحتوي على ستة عشر رمزا .العشر أرقام الأولى *(0 - 9)* زائد الست أحرف الأولى (A-F) . مما يجعل الأمر أسهل نوعا ما.

2- *لغة الأسمبلي(التجميع)*: هي عبارة عن وسيط بين لغة الإنسان و لغة الألة ,حيث أنها اللغة الأولى التي بدأت الاعتماد على الرموز الهجائية بدلا من النظام الثنائي الرقمي.يقوم المبرمج بتصميم تطبيقات و برامج معينة معتمدا على تركيب هذه اللغة تاركا مهام ترجمة الأوامر أو تحويلها إلى لغة الآلة للمفسر *(اسمبلر)*.و نظرا لكون لغة الأسمبلي قريبة من لغة الآلة فإن المبرمج يستطيع الاستغلال الأمثل لموارد الحاسب. حيث أن برامج لغة الأسمبلي تتميز بسرعة التنفيذ و قلة الحجم لذا هي اللغة المفضلة لدى الهاكرز و الكراكرز .

3- *لغات البرمجة العليا:* تشكل هذه اللغات قفزة نوعية في عالم البرمجة , حيث لم يعد لزاما على المبرمج أن يكون على علم بمواقع التخزين و تفاصيل الجهاز.يمكن للبرامج التي تصمم بهذه اللغة العمل على مختلف الأجهزة.كما أن تركيبها بات أكثر يسرا من لغات الأسمبلي كون أن العبارات المستخدمة فيها هي أقرب للغة الإنسان من لغة الآلة.توجد الكثير من لغات البرمجة العليا, فمنها الكوبول و الفورتران و السي و السي ++ , البيسك و باسكال.

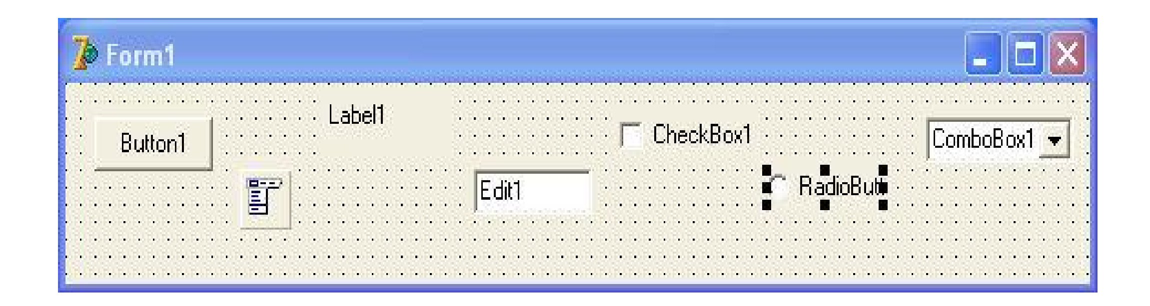
على الرغم من السهولة التي تميزت بها هذه اللغات بالمقارنة لما سبقها, إلا أنها ظلت مقصورة على فئة معينة , باعتبار أن تصميم أي برنامج سيستغرق وقت طويل. فمثلا لإظهار نافذة فارغة كان ينبغي على المبرمج كتابة صفحة أو اثنتين من الأكواد. لقد دفع هذا الأمر شركات البرمجة لتطوير برامج من أجل تسهيل العمل على المطورين, فقد تم دمج بعض لغات البرمجة ضمن بيئات متكاملة تحتوي على أدوات مرئية و هذا من أجل تطوير البرامج بشكل سريع. فنجد أن لغة البيسك تم تطويرها إلى فيزوال بيسك, أما الباسكال فقد ظهر عنها الدلفي الذي نحن بصدد العمل عليه في هذا الكتاب. حيث يتميز الدلفي بالخاصية *RAD* و التي تعني*(Rapid Application Developmen)* أي التطوير السريع للبرمجيات.

إن البرمجة لا تقتصر على ما سبق ذكره فحسب, بل توجد الكثير من لغات البرمجة في عالم الكمبيوتر . نجد برمجة قواعد البيانات و التي تعتمد على لغة الإستعلام البنيوية (SQL) و هو إختصار

ل: (strucured query language) . كما نجد لغات برمجة ويب أو ما يسمى scripting languages مثل لغة HTML , javascript, PHP , ASP , ASP.net . وهي لغات برمجة لتطوير مواقع الإنترنت.
هنالك أيضا لغات برمجة خاصة ببعض البرامج مثل لغة cold fusion و action script الخاصة بالفلاش بلاير.
عموما, يمكن القول أن تعلم لغة برمجة واحدة يمكن أن يكوّن فكرا برمجيا لدى المتعلّم . فمعظم لغات البرمجة تشترك في بعض المفاهيم مثل الكلمات المحجوزة, المتغيرات,الثوابت, الدوال الشرطية و الكثير من الأمور الأخرى.

مقدمة للغة البرمجة دلفي : إن لغة *visual basic* تعلن من تسميتها أنها لغة البيسك المرئية أو الرسومية, بينما التسمية الغير معلنة للدلفي هي باسكال المرئية. و المقصود بهذا أن بيئة التطوير دلفى تعتمد في الأساس على تركيب لغة باسكال مصحوبة بمكتبة ضخمة من الأدوات المرئية الجاهزة و التي يمكن للمبرمج استخدامها مباشرة. و هذه الأدوات في معظم الأحوال لا يخل منها برنامج من البرامج.فلا يمكن أن نتصور برنامج بدون أزرار أو نموذج إلا فيما ندر.

كما سبق الذكر فإن لغة دلفي هي لغة التطوير السريع للبرامج باستخدام الأدوات أو المكونات المتوفرة مثل الأزرار و النموذج و أداة النص و الكثير من الأدوات الأخرى .يعتبر النموذج هو الحاضن الأساسي لهذه المكونات حيث يتم تنسيقها و فق نمط معين يتناسب و العمل المنشود.يعبر عن الأدوات بعدة تسميات منها المكونات أو الكائنات.إن التسمية الأخيرة تبدو أكثر بلاغة, حيث نفهم من ذلك أن كل كائن له مجموعة من الخصائص , من طول و عرض , مسطح أو ثلاثي الأبعاد , مرئي أو غير مرئي, له عنوان معين و إسم يختلف بالضرورة عن باقي المكونات التي تشترك معه في الجنس. على سبيل المثال يمكن أن نعطي تسمية لزر معين Button1 كما يمكن أن نسمي النموذج أيضا بنفس التسمية لاختالفهما في الجنس.بينما لو أضفنا زر أخر و أعطيناه نفس التسمية (Button1) فإن برنامج دلفي يرفضها لأنهما من نفس الجنس.



تمثل هذه الصورة مجموعة من مكونات دلفي ضمن نموذج رئيسي.كما هو واضح من الصورة فإن كل مكون يحمل التسمية الخاصة به, حتى النموذج يعتبر مكون و هو الذي يشكل واجهة المستخدم. من الآن و صاعدا فإن التسميات الخاصة بمكونات دلفي يجب أن تحفظ في الذهن بلغتها الأم .فإذا قلنا الأداة *ComboBox* يذهب ذهن المبرمج مباشرة إلى تلكم الأداة التي تحمل نفس التسمية في الصورة.

بالنسبة لمفهوم الخصائص سيتضح لو قمنا بالتلاعب ببعض المكونات.فلو قمنا بتغيير عنوان النموذج *(caption)* إلى *" البرنامج الأول "* بدلا من *"form1"* نكون قد تعاملنا مع خصائص هذا الكائن و كذلك نفس الشيء لو قمنا بإعطاء مساحة أكبر للأداة *Edit1* .إن عمل هذه المكونات يتوقف على مجموعة من الأكواد و الإجراءات *(procedures)* التي تنفذ إثر القيام بحدث معين *(event)* لمكون ما .فالنقر مثلا يعتبر حدث و تمرير الفأرة على أداة معينة يعتبر كذلك حدثا . فيمكننا مثلا إعطاء أمر بإغلاق البرنامج إثر تنفيذ حدث تمرير الفأرة على الأداة *CheckBox1*. و هكذا لو قام المستخدم بتمرير الفأرة على هذه الأداة فإن البرنامج سوف يغلق مباشرة.

تمر البرمجة بدلفي بمرحلتين أساسيتين , يتم في المرحلة الأولى تنسيق و ضبط خصائص المكونات على النموذج.بينما يتم في المرحلة الثانية كتابة أكواد برمجية ضمن إجراءات مرتبطة بأحداث معينة.

عند فتح برنامج دلفي فإنه يقوم تلقائيا بإنشاء نموذج يحمل الخصائص الإفتراضية و التي يمكن تغييرها من قبل المبرمج.فإذا عرفنا سابقا أن المرحلة الأولى تتم على النموذج *فأين يا ترى يتم كتابة الإجراءات؟.*

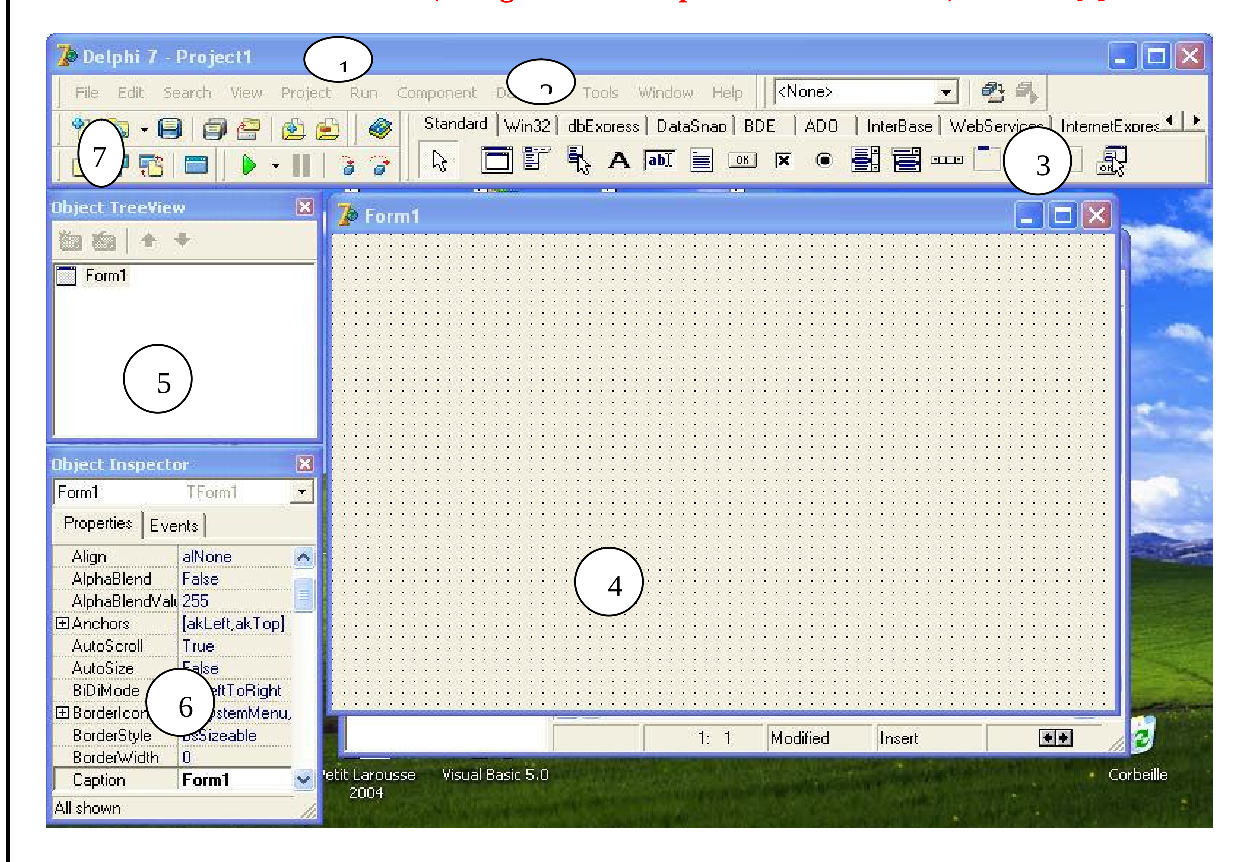
مثلما يقوم بإنشاء النموذج بشكل تلقائي, فإن دلفي يقوم بنفس العمل مع الوحدات units و التي يتم فيها كتابة الإجراءات و الوظائف ,مع العلم أن كل نموذج ينفرد بوحدة خاصة به .
عادة ما تكتب الأوامر بين الكلمتين المحجوزتين begin و end .

```
procedure Tform1.Button1Click(Sender: TObject);
begin
هنا تكتب الأوامر
end;

end.
```

السطر الأول يضم إسم الإجراء وهو الإجراء المرتبط بحدث النقر على الأداة button1.
أما السطر الأخير فهو يعلن عن نهاية كل الإجراءات حيث ما يتم كتابته بعد هذا الأمر سيتم تجاهله أثناء تنفيذ البرنامج. و معنى التنفيذ هو تشغيل البرنامج بعد الإنتهاء من برمجته جزئيا أو كليا. أي بعد الإنتهاء من وضع التصميم تأتي مرحلة التنفيذ و التي من خلالها نكتشف ما إذا ارتكبنا أخطاء برمجية ما .لأن دلفي عادة ما يفضح المبرمجين في هذه المرحلة(مجرد مزحة ,أقصد ينبههم لوجود لبس ما في الكود) قد يبدو الجانب النظري صعب الفهم نوعا ما لكننا بمجرد البدء في العمل التطبيقي ستتضح الأمور.

بيئة التطوير المتكاملة (Integrated Development Environment)



تمثل هذه الصورة الواجهة الرئيسية لبيئة التطوير المتكاملة الخاصة بلغة دلفي.حيث تحتوي على كل ما يحتاجه المبرمج من أدوات لبناء برامجه و تجريبه . و يأتي مع بعض نسخ دلفي برنامج INSTALL SHIELD الذي يسمح بتحزيم أو عمل setup للبرنامج الذي تم تصميمه, وهذا البرنامج يأتي منفصل عن بيئة التطوير المتكاملة.لكن لن تحتاج إلى هذا البرنامج لتوزيع برنامجك في كل الأحوال.لأن دلفي يدعم

خاصية stand alone مما يجعل المبرمج يشغل برنامجه خصوصا إذا كان صغير الحجم دون الحاجة لتحميل أي ملف إلى جهاز الزبون.و كي نفهم خاصية stand alone يمكن أن نعطي مثالا عن الفاكس فهذا الجهاز لا يحتاج لأي جهاز آخر لكي يعمل, بينما الطابعة لا تدعم هذه الخاصية لأنها تحتاج لجهاز كمبيوتر لكي تعمل.هذا المثال يتعلق بالعتاد و يمكن تفسيره على SOFTWARE.مع هذا يجب أن لا نعتقد بأن الدلفي لن يحتاج لأي ملف مرفق لكي يعمل, بل سيحتاج المبرمج إلى عمل setup لبرنامجه لكي يرفق الكثير من الملفات, ففي أبسط الأحوال ستحتاج لتحزيم بعض ملفات الخطوط مع برنامجك و التي يعتقد أنها غير موجودة عند جهاز الزبون, فضلا عن الملفات الأساسية التي بدونها لن يعمل البرنامج بشكل ملائم أو لا يعمل على الإطلاق.

عموما يمكن تقسيم بيئة التطوير دلفي إلى 6 أجزاء كما في الصورة:

	العنوان	**المهام**
1	title bar	أو شريط العنوان حيث يظهر فيه إسم المشروع بعد رقم إصدار دلفي, و يمكن تغيير إسم المشروع عند عملية التخزين.
2	main menu	أو شريط القوائم و يضم مجموعة من القوائم منها ما هو شائع في مختلف البرامج مثل القائمة FILE و EDIT و منها ما هو خاص ببيئة دلفي:

file	تستخدم لفتح و إعادة فتح و كذا حفظ المشاريع و إغلاق بيئة التطوير.
edit	تضم بعض الأوامر الهامة كالتراجع عن آخر عمل قمت به,فضلا عن أوامر النسخ و اللصق و التنسيق.
search	للقيام بعملية البحث عن أي كائن أو عبارة داخل البرنامج
View	تستخدم لإظهار بعض النوافذ في بيئة التطوير.
project	لإدارة المشروع و بناءه فضلا عن إستيراد و إضافة مكتبات أخرى و كذا ضبط خصائصه من حيث ظهور النماذج و الأيقونة.
run	لتنفيذ البرنامج, و يمكن عمل ذلك من خلال المؤشر الأخضر الذي يظهر في الصورة أعلاه.كما تتميز هذه القائمة بتوفر عدة أوامر للتعامل مع تنقيح البرنامج أو ما يعرف بـ debugging .
Component	للتعامل مع المكونات كإضافة أدوات جديدة غير موجودة ضمن قائمة الأدوات, أو العمل على إنشاءها.
Database	تستخدم للتعامل مع الأدوات المرتبطة بقواعد البيانات
Tools	تظم قائمة من الأدوات و البرامج المنفصلة و التي يمكن استدعائها من هذه القائمة لتسهيل المهام على المبرمج
Windows	من القوائم الشائعة في مختلف البرامج و وظيفتها التحكم في ظهور النوافذ المفتوحة داخل بيئة دلفي.
Help	تعتبر إحدى أهم القوائم لاحتوائها على ملفات المساعدة الخاصة بلغة دلفي و بيئة التطوير المتكاملة.كما أنها تحتوي على الأمر about و الذي يمكّن المبرمج من معرفة تفاصيل إصدار الدلفي التي يعمل عليها.كما نجد قسما مخصصا لبعض المواقع الخاصة بشركة بورلاند و لغة دلفي بصفة خاصة و التي

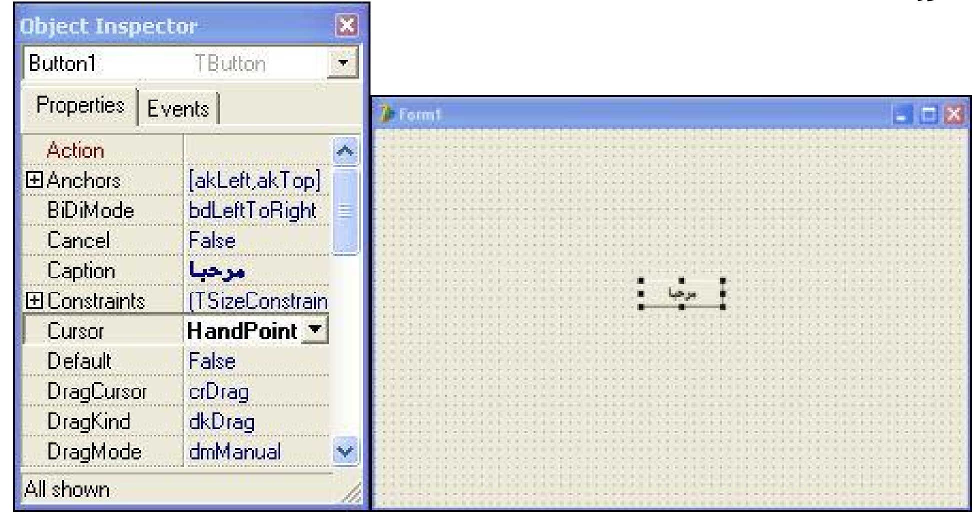
		يمكن للمبرمج زيارتها خصوصا إذا أراد مطالعة الأسئلة الشائعة عن دلفي.
3	شريط الأدوات أو مكتبة المكونات المرئية " VCL" Visual component Library	و تحتوي على مكتبة ضخمة من الأدوات المرئية التي تميز لغة دلفي عن باقي لغات البرمجة.يمكننا تصنيف هذه الأدوات حسب وظيفتها إلى أبواب حيث أن كل باب يحتوي على مجموعة من الأدوات الخاصة به. يمكننا بصفة عامة عرض الأبواب الأساسية في هذا الجدول. **أبواب الأدوات الأكثر إستعمالا**: نجد في القائمة standard الأدوات القياسية و التي لا يمكننا الإستغناء عنها, مثل الأداة Panel,Label,ComboBox,Button.كما نجد قوائم أخرى مثل : Win3.1,Additional, Win32,Sample,System, dialogs . و التي تتعدى مهام الأدوات الموجودة فيها من أدوات تحاكي أنظمة التشغيل إلى أدوات للتعامل مع صناديق الحوار , فضلا عن أدوات التنسيق. **أبواب أدوات قواعد البيانات**: توجد الكثير من الأدوات للتعامل مع قواعد البيانات من حيث الإتصال و إنشاء الإستعلامات و التي يمكن الوصول إليها من خلال عدة قوائم: ADO,DBE,DataSnap,Dbexpress,Interbase, Data access, Data contorols. و تضم هذه الأخيرة الأدوات الأكثر إستعمالا في قواعد البيانات مثل: DBGrid,DBNavigator,DBEdit **أبواب أدوات التعامل مع الويب**: و تتنوع فيها الأدوات حسب التبويب التي تنضوي تحته و إن كانت في مجملها تصب في مفهوم برمجة تطبيقات ويب و التعامل مع المزود و الزبون .أهم الأبواب المؤدية لهذه الأدوات هي : WebServices,InternetExpress,Internet, WebSnap,
		بالنسبة للأدوات التي يمكن إستدعائها من القائمة component في شريط القوائم, فهي عادة ما تظهر في التبويب ActiveX.
4	**النموذج** (Form)	و هو الحاضنة التي تجمع الأدوات المستخدمة في المشروع, حيث أنه يشكل واجهة التطبيق .
5	Object TreeView	يقوم هذا الإيطار بعرض الأدوات المستخدمة في المشروع بشكل علائقي. حيث يمكن للمستخدم مشاهدة الأدوات و تحديدها من أجل ضبط خصائصها من نافذة الخصائص.
6	Object Inspector	يحتوي هذا الإيطار على صفحتين أساسيتين : Properties: و تضم خصائص كل أداة مستخدمة, حيث يمكن للمبرمج التعديل عليها. Events: صفحة الأحداث تمكننا من إختيار الحدث المرتبط الذي نريد إنشاء إجراء إثر حدوثه, يمكن إختيار أي حدث بالنقر المزدوج على الإيطار المقابل له, و سوف تفتح نافذة تحرير الكود مصحوبة بالإجراء المرتبط بالحدث المحدد.
7	**Speed Buttons Bars**	تضم هذه الأشرطة أزرارا للوصول السريع لأهم أوامر دلفي .يمكن تنفيذ البرنامج أو إضافة نموذج من خلال هذا الشريط. يجدر بنا الذكر أن أسماء و وظائف أدوات دلفي عادة ما تظهر عند وضع المؤشر عليها.فلو قمت مثلا بوضع المؤشر فوق الزر الأخضر, سيظهر التعليق التالي "Run F9" . و الذي يخبرنا بمهام الزر المحدد.
8	UNIT	تظهر هذه النافذة خلف النموذج , و تستخدم كمحرر للأكواد .و كما ستلاحظ فهي تتكون من التركيب الخاص بلغة دلفي حيث تبدأ بإسم الوحدة و تنتهي بالكلمة End

لا يمكننا فهم الدور الحقيقي للعناصر السابقة الذكر ما لم نقم بتجريب العمل عليها.لهذا سنستعجل قليلا لإنشاء أول مشروع بلغة دلفي.

أول مشروع بلغة دلفي:

عند تشغيل دلفي فإنه يقوم تلقائيا بإنشاء نموذج كي يحوي الأدوات التي تود استخدامها.لهذا ما يجب عليك القيام به هو وضع الأداة Button على النموذج ,حيث يمكن الوصول لهذه الأداة من قائمة الأدوات Standard .

قم بتحديد هذه الأداة بوضع مؤشر الفأرة عليها ثم انتقل إلى نافذة الخصائص لضبط خصائصها كما في الصورة.



يمكن ملاحظة أن إسم الأداة في أعلى نافذة الخصائص و هو "Button1" .أما عنوان الأداة أي "caption" فهو " مرحبا". بالنسبة للخاصية Cursor فهي " crHand point ". لاحظ أن عنوان الأداة يظهر مباشرة بعد القيام بتعديله حيث كان في السابق Button1 , أما الخاصية الثانية التي قمنا بتعديلها Cursor فهي لتحديد نوع المؤشر الذي يظهر على الأداة عند تقريب مؤشر الفأرة منها.و كما تشاهد في القائمة, فقد اخترنا مؤشر اليد .لكن هذه الخاصية لا يمكن ملاحظتها أثناء التصميم بل لابد أن ننتظر وقت التنفيذ.توجد الكثير من الخصائص التي يمكن التعديل فيها, لكننا مع ذلك لن نثقل عليك في مشروعك الأول.

الآن حان وقت كتابة الأكواد.حسنا, قم بالنقر المزدوج على الزر " مرحبا" في النموذج أو على الأداة Button1 في إيطار Object TreeView ليظهر لك محرر الأكواد.
بين الكلمتين المحجوزتين Begin و End أكتب الأمر الذي يظهر باللون الأزرق:

```
procedure Tform1.Button1Click(Sender: TObject);
begin
ShowMessage( 'مرحبا بكم في عالم دلفي' );
end;

end.
```

مفاد الأمر السابق هو :عند النقر على الأداة Button1 , أظهر الرسالة التالية " مرحبا بكم في عالم دلفي".
للتأكد من صحة ما قمت به , يجب تنفيذ البرنامج و ذلك إما من القائمة Run ضمن شريط القوائم أو السهم الأخضر ضمن Speed Buttons Bars , أو بالضغط على F9. قم بالضغط على الأداة التي تحمل العنوان" مرحبا " و سيظهر ما يلي:



إذا نجحت العملية فهذا أمر جيد. إذا لم تنجح قم بمراجعة الكود ,فلعلك نسيت قوسا أو حرفا ما ضمن الكود المستخدم .فلو قمت بكتابة showmesage فإن تنفيذ البرنامج لن ينجح لأنك ارتكبت خطأ املائيا.
بعد نجاح العملية, قم بإغلاق البرنامج من زر الإغلاق أو من خلال *Program Reset* في قائمة *Run* , و توجه مباشرة إلى القائمة *File* و اختر *Save project as* من أجل حفظ المشروع .يستحسن أن تنشأ مجلد مسبقا للمشروع و ليكن في القرص *C* تحت عنوان *Program*. طبعا لك الحرية في إسم المجلد و مكانه و لكن ما اقترحته لتسهيل الوصول للمجلد فقط.حسنا بعد قيامك باختيار *Save Project As* قم بتغيير إسم الوحدة من *Unit1* إلى *HELLOF* ثم قم بالحفظ في المجلد المحدد .ثم قم بتغيير إسم ملف المشروع من *Project1* إلى *HELLO*. قبل إغلاق بيئة دلفي قم بفتح المجلد الذي حفظت به الملفات السابقة.حيث ستجد ست ملفات. لكن أين الملف التنفيذي الذي يحمل الإمتداد EXE؟.
لا عليك قم بتنفيذ البرنامج الذي صممته مجددا من بيئة دلفي , ثم قم بإغلاقه و إغلاق بيئة دلفي ككل, و انتقل إلى المجلد و ستجد الملف التنفيذي , حيث يمكنك النقر عليه بشكل مزدوج ليعمل خارج بيئة دلفي.
إن ما قمت به هو مجرد برنامج صغير يمكن أن تنقله لجهاز آخر لا يحتوي على بيئة دلفي, و سوف تلاحظ أنه سيعمل دون أي مشكلة.طبعا يمكنك أن تبني برامج محترمة و توزيعها بهذا الشكل لكن ينبغي أن تكون ملما ببعض الأمور الهامة التي تساعدك في بناء برامجك , مثل التركيب الداخلي للغة دلفي و كذا كيفية إنشاء واجهة مستخدم سهلة و أنيقة في نفس الوقت.
في الخطوات القادمة لن نكرر ما ذكرناه سابقا مثل كيفية تنفيذ المشروع أو حفظه أو إضافة أدوات إلى النموذج .إلا إذا دعت الحاجة إلى ذلك .
ملاحظة هامة: إن كل أداة في لغة دلفي تحمل إسم خاص بها يظهر في نافذة الخصائص على الشكل Name. كما أن لمعظم الأدوات عنوان Caption . العنوان يظهر على الأداة بمجرد تعديلها.بينما الإسم يقتصر إستخدامه ضمن الأكواد البرمجية.ببساطة يمكن أن تكتب عنوانا باللغة العربية كما عملنا سابقا (مرحبا) بينما لا يمكن استخدام إسم باللغة العربية.
بعض الأدوات لا تحتوي على الخاصية Caption حيث ينوب عنها الخاصية Text أو أمور أخرى, كما هو الحال للأداة Edit.

الخصائص و الأحداث

1- الخصائص

تشترك الكثير من الأدوات في بعض الخصائص التي لابد من معرفتها. بينما تبقى بعض الخصائص حكرا على أداة دون أخرى.
في الشكل التالي سنقوم بالتعرف على أهم الخصائص. و نأخذ الأداة Form كمثال لعملنا هذا , بينما يستطيع كل شخص اختبار خصائص باقي الأدوات بمفرده.
طبعا لا يسعنا الوقت للحديث عن جميع الخصائص و إنما سنركز على الأهم فالأهم.

Form1	TForm1
Properties	Events
Action	
ActiveControl	
Align	alNone
AlphaBlend	False
AlphaBlendVal	255
⊞Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞BorderIcons	[biSystemMenu
BorderStyle	bsSizeable
BorderWidth	0
Caption	**Form1**
ClientHeight	334
ClientWidth	529
Color	**clBtnFace**
⊞Constraints	(TSizeConstrai
Ctl3D	True
Cursor	crDefault
DefaultMonitor	dmActiveForm
DockSite	False
DragKind	dkDrag
DragMode	dmManual
Enabled	True
⊞Font	**(TFont)**
FormStyle	fsNormal
Height	**368**
HelpContext	0
HelpFile	
HelpKeyword	
HelpType	htContext
Hint	
⊞HorzScrollBar	(TControlScroll
Icon	(None)
KeyPreview	False
Left	**192**
Menu	
Name	Form1
ObjectMenuIte	
OldCreateOrde	**False**
ParentBiDiMod	True
ParentFont	False
PixelsPerInch	96
PopupMenu	
Position	poDesigned
PrintScale	poProportional
Scaled	True
ScreenSnap	False
ShowHint	False
SnapBuffer	10
Tag	0
Top	**114**
TransparentCol	False
TransparentCol	clBlack
UseDockMana	False
⊞VertScrollBar	(TControlScroll
Visible	False
Width	**537**
WindowMenu	
WindowState	wsNormal

الخاصية	الشرح
Align	لجعل المحاذاة لليمين أو اليسار . أعلى أو أسفل أو تأخذ جميع المساحة. على أية حال في حالة النموذج ستنطبق المحاذاة على الشاشة.بينما لو و ضعت أداة ما على نموذج و ضبطت المحاذاة فستكون هذه المحاذاة بالنسبة للنموذج.
AutoScroll	لتمكين شريط التمرير في النموذج.مثلا قم بوضع أداة ما و اعطها مساحة أكبر من مساحة النموذج. و ستلاحظ ظهور شريط التمرير .يمكنك تعطيل الخاصية باختيار False
BiDiMode	لتغيير الإتجاه من اليسار إلى اليمين . هذه الخاصية مهمة جدا في البرامج الموجهة للسوق العربية.
BorderIcons	لتمكين أو تعطيل خاصية التكبير و التصغير لواجهة النموذج.(في بعض الأحيان تحتاج أن يبقى برنامجك على مساحة معينة)
BorderStyle	للتعامل مع الحافة العليا للنموذج.
Caption	عنوان الأداة
Color	لون النموذج
Cursor	نوع المؤشر
Enable	تمكين التعامل مع النموذج من قبل المستخدم أو تعطيله.
Font	نوع الخط, لونه و حجمه
FormStyle	الخاصية fsStayOnTop تجعل البرنامج دوما في المقدمة
Hint	لإضهار تلميحات.سبق و أن ذكرنا أنه بمجرد وضع الفأرة على أحد أدوات دلفي إلا و يظهر تلميح بمهامها. حسنا يمكنك فعل نفس الشيء شريطة ضبط خاصية ShowHint على True
Icon	لاختيار أيقونة للبرنامج
Name	إسم الأداة
Position	موضع النموذج من الشاشة
ShowHint	لتمكين أو تعطيل التلميحات
Visible	لتمكين رؤية الأداة أو تعطيلها
WindowState	لتحديد حالة نافذة البرنامج.إما تشغل كامل الشاشة أو أن تفتح على حالة التصغير.كما يوفر دلفي إمكانية ترك النافذة على الحالة التي صممت عليها
Height/Width	للتعامل مع مساحة النموذج
Horz/vertScrollBar	للتعامل مع شريط التمرير الأفقي و العمودي.
	يمكنك تجربة كل هذه الخصائص لمعرفة عملها بشكل واضح.و من المؤكد أنها جد مفيدة للمبرمج في التطوير السريع لبرنامجه. عموما, يمكن ضبط معظم الخصائص برمجيا. من خلال بعض الأكواد.فمثلا لو أردنا أن نغير خاصية عنوان النموذج برمجيا فهذا الكود سيكون كفيلا بالعملية Form1.caption:=' PROG ' يمكن جعل الكود يعمل مع بداية البرنامج لو اخترنا الحدث OnCreate و الذي ستتعرف عليه ضمن قائمة الأحداث التالية.

2- الأحداث

يمكنك إختيار الحدث المناسب من خلال القائمة التالية و ذلك بالنقر المزدوج على الإيطار المقابل لها, لتفتح نافذة تحرير الأكواد. سنأخذ الكود السابق الذي يقوم بعرض رسالة للمستخدم مع الأحداث التالية كي نفهم جيدا وضيفتها

اسم الحدث	الشرح
OnActivate	حدث تنشيط النموذج
OnClick	حدث النقر
OnClose	حدث غلق النموذج
Oncreate	حدث إنشاء أو فتح البرنامج
OnDblClick	حدث النقر المزدوج
OnDeactivate	حدث عدم التنشيط
OnKeyPress	حدث الضغط على لوح المفاتيح(في بعض الأحيان تقوم بادراج عبارة في محرك بحث لبرنامج أو موقع ما و تقوم بالضغط على الزر Entr لتبدأ عملية البحث, أي دون إستخدام الفأرة. يمكنك فعل نفس الشيء, و هذا الحدث جد مفيد في حالة التعامل السريع مع أداة ما.
OnMouseMove	حدث تمرير الفأرة
OnResize	حدث تكبير أو تصغير النموذج
تلميح	**توجد بعض الأحداث في أدوات و لا توجد في أخرى. فمثلا الأداة** WebBrowse **تحتوي على الكثير من الأحداث الغير متواجدة في الأدوات الأخرى. بمرور الوقت يمكن للمبرمج اكتساب خبرة أكبر في التعامل مع الأحداث .**
مثال	قم بالنقر المزدوج على الحدث OnClose و اكتب الكود التالي: ***ShowMessage('إلى اللقاء');*** قم بتجربة البرنامج. حسنا, قم بإغلاقه و ستلاحظ ظهور الرسالة.إن ما قمنا به هو مايلي. *عند إغلاق النموذج* (هذا يمثل الحدث) *أظهر الرسالة ' إلى اللقاء'* (و هذا هو الكود) يمكن تنفيذ نفس الشيئ مع باقي الأحداث . فكرة الأحداث جميلة جدا. وهي تساعدك على التعامل مع المستخدم .تستطيع استغلالها لافتتاح برنامجك بملف صوتي معين (آية أو سورة من القرآن الكريم....) .و انهائه بنفس الشيء, لكن يرجى الإنتباه لمثل هذا الإجراء لأنه في بعض الأحيان يأخر فتح البرنامج مما يسبب استياء لدى المستخدم. خصوصا المستخدم العربي فهو قصير النفس .لذا هو لا يحتمل أن تأخره بحدث ما مع كل تشغيل للبرنامج.

Form1 TForm1
Properties Events
Action
ActiveControl
Menu
ObjectMenuIter
OnActivate
OnCanResize
OnClick
OnClose
OnCloseQuery
OnConstrainedI
OnContextPopu
OnCreate
OnDblClick
OnDeactivate
OnDestroy
OnDockDrop
OnDockOver
OnDragDrop
OnDragOver
OnEndDock
OnGetSiteInfo
OnHelp
OnHide
OnKeyDown
OnKeyPress
OnKeyUp
OnMouseDowr
OnMouseMove
OnMouseUp
OnMouseWhee
OnMouseWhee
OnMouseWhee
OnPaint
OnResize
OnShortCut
OnShow
OnStartDock
OnUnDock
PopupMenu
WindowMenu
All shown

مشاريع عملية

بعدما تعرفنا على الخصائص و الأحداث , ننتقل الآن لإنجاز بعض المشاريع العملية التي تساعدنا على فهم التركيب الداخلي للغة باسكال.نحن في هذا الكتيب لن نتعمق في فهم تركيب هذه اللغة و إنما سنكتفي ببعض النماذج التي تجعلنا نعتاد على التعامل معها في معظم المشاريع.
إذا كنت ترغب في فهم كل صغيرة و كبيرة عن لغة باسكال المدمجة في لغة دلفي فإننا ننصحك بمطالعة الكتب المتوفرة في السوق أو على الشبكة المعلوماتية(سنخصص في آخر الكتاب فسحة للتعرف على الكتب و المواقع المهتمة بهذه اللغة).

بداية يمكن القول أن معظم اللغات تحتوي على الدوال الشرطية و التي تساعدنا على القيام بحدث وفقا لتحقق الشرط من عدمه.
مثلا في بعض البرامج يطلب منك إدراج كلمة المرور , فإذا كانت الكلمة صحيحة يفتح البرنامج أما إذا كانت خاطئة فيغلق البرنامج.
سنقوم في المشروع التالي بنفس العمل .حيث سيطلب من المستخدم إدراج كلمة مرور نحن نحددها.إذا توافقت كلمة المرور المدرجة في الإيطار المخصص مع الكلمة التي حددناها سنقوم بتغيير عنوان النموذج إلى عبارة ' هذا البرنامج مسجل'
أما إذا كانت كلمة المرور خاطئة فإننا نغير عنوان النموذج إلى عبارة ' غير مسجل'
الصورة التالية توضح ما سبق ذكره.



لا حظ أن النموذج في الخلف لا يحتوي إلا على أداة و احدة هي BUTTON .
قم بانشاء مشروع جديد و ضع الأداة BUTTON على النموذج, ثم انقر نقرا مزدوجا عليها و اكتب الكود التالي:

```
procedure TForm1.Button1Click(Sender: TObject);
 var
 InputString: string;
begin
 InputString:= InputBox ( 'كلمة المرور' ,'أدخل كلمة المرور', ' ' );
 if InputString = 'DELPHI' then
 Form1.Caption:=( ' هذا البرنامج مسجل ' )
 else
 Form1.Caption:=('غير مسجل');
 end;
```

قم بكتابة الكود الظاهر باللون الأزرق لأن باقي العبارات تدرج من قبل دلفي تلقائيا.
سنتعرف الآن على الكود خطوة خطوة.كي يكون باستطاعتك انشاء نماذج مماثلة لا حقا.
قمنا أولا بالإعلان عن وجود متغير ' var '
اسم المتغير هو ' InputSTRING ' و هو من نوع STRING أي سلسلة نصية

حيث يمكن أن يكون المتغير من نوع حرف *CHAR* , سلسلة نصية *STRING* , بولين *BOOLEAN* .عدد صحيح *INTEGER*...
بالنسبة لإسم المتغيير , يمكنك استبداله بأي تسمية أخرى .
بعد العبارة *begin* التي تعني بداية تنفيذ الكود, نجد أننا قد أسندنا قيمة للمتغير وهي التي تظهر بعد المساواة. وهذه القيمة يمكن إستبدالها و فق الحاجة. ما يظهر بعد المساواة هو عبارة عن صندوق الحوار الذي يظهر في الصورة.حيث أن كلمة المرور تكون عنوانا لصندوق الحوار أما العبارة التي تليها فهي لتحديد الطلب.في حين أن المزدوجتين التي تظهر في آخر السطر تترك شاغرة دون أي مساحة لإنها مخصصة للمستخدم لأدراج كلمة المرور.
لاحظ أنني قمت بترك مساحة بين المزدوجتين كي يفرق القارئ لهذا الكتاب أنها ليست الرمز (").
باعتبار أنني أستخدم معالج النصوص *MICROSOFT WORD* في تحرير هذا الكتيب.
السطر الموالي يحتوي على البنية الشرطية و التي تبتدأ بالكلمة IF و تنتهي في نفس السطر ب الكلمة المحجوزة Then
مفاد الأمر السابق : **إذا** كان النص المدرج في صندوق الحوار **يساوي** الكلمة ' DELPHI ' **إذن** حقق الشرط الموالي , أي اجعل عنوان النموذج هو ' هذا البرنامج مسجل'
عدا ذلك (أي إذا كانت الكلمة المدرجة من قبل المستخدم ليست (' DELPHI ')
إجعل عنوان النموذج ' غير مسجل'.

IF	إذا
THEN	إذن / بالتالي
Else	عدا ذلك (عدم تحقق الشرط)

نفذ البرنامج و قم بإدخال كلمة المرور الصحيحة (DELPHI) و ستلاحظ تحول عنوان النموذج إلى مايلي:



قم بتجريب كلمة أخرى و ستلاحظ تحول العنوان إلى ما يلي:



نحن في مثالنا هذا ربطنا نتيجة عدم تحقق الشرط بتحويل عنوان النموذج إلى ' غير مسجل'.
يمكنك استبدال هذا الشرط بشرط آخر.
قم بحذف آخر سطر في الكود

```
Form1.Caption:=('غير مسجل');
```

و استبداله بالأمر التالي:

```
Close;
```

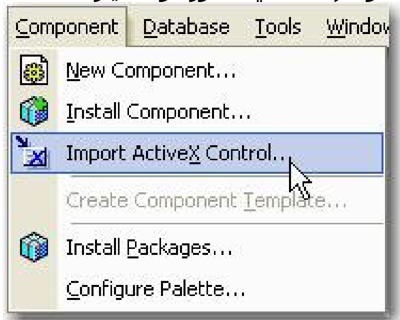
و ستلاحظ انهاء البرنامج في حالة عدم تحقق الشرط.

كان هذا مجرد مثال بسيط عن الدالة الشرطية *IF* و ستلاحظ عدم كفائتها في بعض الأحوال. لكن لا عليك فلغة دلفي توفر لك الكثير من البدائل.
من بين الدوال الشرطية الأكثر شهرة في لغة دلفي نجد الدالة *Case* . و التي سنستثمرها في مشروعنا التالي.
المشروع التالي هو قاموس إنجليزي عربي ناطق.
طبعا لن نعتمد على قواعد بيانات في هذا المشروع نظرا لتطلبها للكثير من الوقت و التركيز.
لكن قبل ذلك خذ لك نفسا. قبل المتابعة.

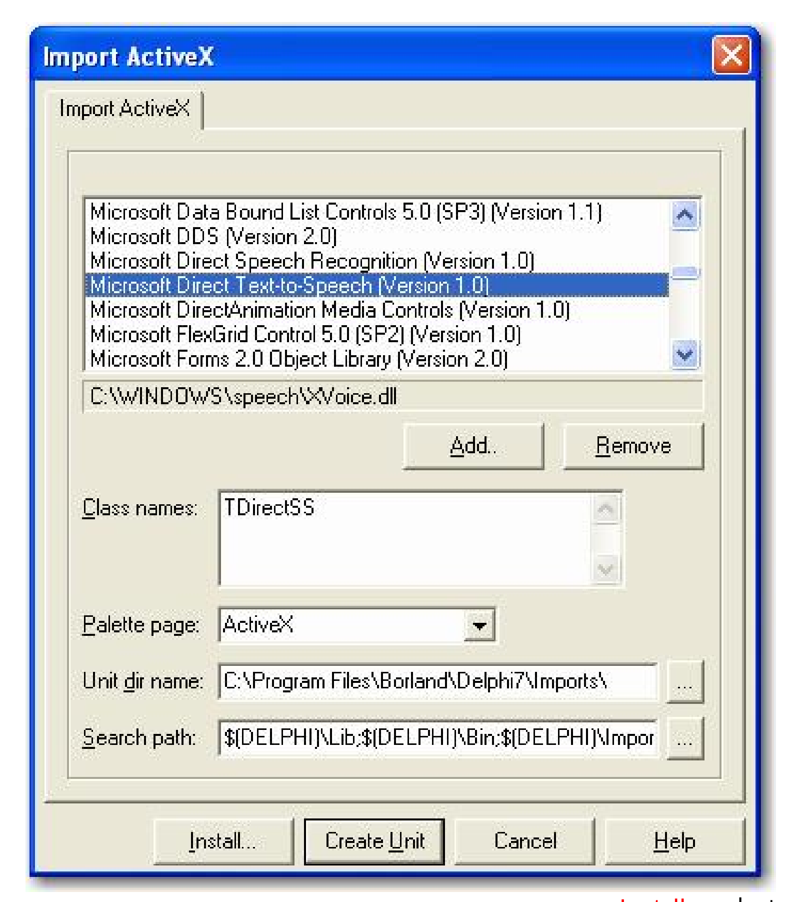
القاموس الإنجليزي العربي الناطق:

في هذا المشروع يرجى الإنتباه للدالة الشرطية فهي الأساس في مشروعنا هذا.
قبل البدء لا بد و أن يكون برنامج *Microsoft text to speech* مثبتا على جهازك لاضافة خاصية النطق للبرنامج كما و لابد من توفر نفس البرنامج على جهاز الزبون لكي يعمل بنجاح.
يمكنك الحصول على البرنامج من موقع الشركة على الشبكة, كما يمكن الحصول عليه ضمن برامج القواميس الناطقة إن كنت تملكها.
إن لم تستطع الحصول على هذا البرنامج , فلا يمكنك الأستفادة من هذه الخاصية.

أولا يجب إضافة الأداة DirectSS للقائمة ActiveX و ذلك بالإنتقال للقائمة Component ضمن شريط القوائم كما في الصورة و اختيار Import ActiveX Control .



قم بتحديد الأداة كما في الصورة..



ثم اختر Install .
بعد ذلك انتقل لقائمة الأدوات ActiveX ضمن شريط الأدوات و ستجد الأداة التي أضفناها كما في الشكل



الآن قم بانشاء مشروع جديد File/new/application
ضع الأدوات التالية على النموذج:
أولا ضع الأداة ControlBar التي يمكن الوصول إليها من قائمة الأدوات Additional و اضبط خصائصها على النحو التالي
Align = alTop
ثانيا,قم بوضع زرين من نوع Button على الأداة controlBar من القائمة Standard, زائد زر آخر من نوع BitBtn من القائمة Additional و اضبط خصائصها على النحو التالي:

Button1	Caption= إستماع	
Button2	Caption= حول	
BitBtn1	Caption= خروج	Kind= bkClose

ضع أداتين من نوع Label و غير تسميتهما كما في الصورة
ضع أداتين من نوع Edit و اجعل الخاصية text فارغة , كما يمكنك تغيير لون الأداتين إلى clInfoBk.
لا تنسى جعل لون الخط هو الأسود من خلال الخاصية font

أضف الأداة ListBox من القائمة Standard واضبط خائصها كما فعلت معا الأداة Edit
إضافة إلى الخاصية التالية
Align=alLeft
أضف الأداة DirectSS من القائمة ActiveX .
واضبط خصائصها كما يلي
Visible=false
أضبط خصائص النموذج على النحو التالي:
BiDiMode= bdRightToLeft
Caption= القاموس الناطق
Color= clMaroon
من التفرع font
Color= clWhite
من التفرع BorderIcons
BiMaximize=false
و سوف يكون النموذج على النحو التالي



كي تظهر الأزرار على الشكل الذي يحاكي نظام Windows Xp يمكنك إضافة الأداة XpManifest من القائمة Win32 . و هذه الأداة متوفرة ابتداءا من الإصدار السابع لبرنامج دلفي.
ما تلاحظ أنه ناقص سوف نقوم بإضافته فيما يلي:
حدد الأداة LisBox
من الخاصية Items قم بتعبئة النافذة التي تظهر كما في الصورة.
انقر نقرا مزدوجا على الأداة ListBox1 و اكتب الأمر التالي بين الكلمتين المحجوزتين Begin و End

```
Case ListBox1.ItemIndex of
0 : edit2.text := 'رأس ' ;
1 : edit2.text :='شعر' ;
2 : edit2.text :='يد' ;
3 : edit2.text := 'أنف'  ;
4 : edit2.text:= 'جبين'  ;
5 : edit2.text:='وجه' ;
6 : edit2.text:='أسنان' ;
7 :edit2.text :='أذن' ;
8 : edit2.text :='عين' ;
9 : edit2.text:=  'خد' ;
end ;
 edit1.text:= ListBox1.Items[ListBox1.ItemIndex] ;
```

مفاد الكود السابق هو في حالة التعامل مع عناصر الأداة ListBox1 فإن الضغط على العنصر الأول في القائمة سينتج عنه تغيير في نص الأداة Edit2 و فق ما يحدده المبرمج.و قس على هذا النحو مع العناصر الموالية.

السطر الأخير في الكود يسمح بإظهار العنصر المحدد في القائمة في أداة النص Edit1 . و الغاية من هذا الأمر هو تمكيننا من تسهيل نطق الكلمة المحددة .

جرب البرنامج و ستلاحظ عملية الترجمة.

بالنسبة لخاصية نطق الكلمات المحددة فهي جد سهلة: قم بالنقر المزدوج على زر الإستماع, و اكتب الكود التالي:

```
DirectSS1.Speak(Edit1.text) ;
```

بالنسبة لزر الخروج فهو لا يحتاج لأي أمر لكن مع ذلك. يمكن استعمال الأمر Close إذا ما دعت الحاجة إلى ذلك.

أما الزر حول البرنامج فيمكنك اظهار رسالة حوار كما فعلنا سابقا.

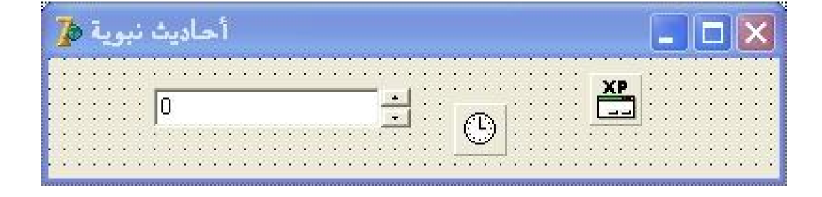
أما إذا أردت أن تجعل النافذة التي تظهر أكثر جاذبية فسيكون بإمكانك إضافة نموذج جديد للبرنامج و قم بإضافة ما تريده من معلومات صور أو روابط لمواقع معينة. ما عليك القيام به بعد ذلك هو مجرد و ضع الكود التالي: إثر حدث النقر على زر حول البرنامج

```
form2.show
```

و الذي يقوم بإظهار النموذج الثاني إذا ما افترضنا أنك قد قمت حقا بإنشائه.

برنامج عارض الأحاديث النبوية

سنقوم الآن بإنشاء برنامج لابد و أن الكثير منا كانت لهم رغبة جمة في بناء برامج مماثلة له. فنجد أن الكثير من المبرمجين يسعون حثيثا لمحاكاة التطبيقات التي تعمل مع بدأ التشغيل مثل برنامج الأذان الشهير. نحن في هذا المشروع لن ننشأ برنامج بحجم هذا الأخير, بل سنكتفي ببرنامج يعمل مع بدء تشغيل الويندوز ليظهر لنا في كل مرة حديثا نبويا (يمكن للمبرمج اختيار ما يشاء) على هيئة صورة jpg بينما يمكن استبدال هذه الهيئة بنص txt أو ملف صوتي mp3 و قس على ذلك. و المقصود من هذا البرنامج هو إنعاش العقل الرياضي للمبرمج. فيا ترى كيف يتم ذلك ؟



كل ما نحتاجه هو الأداة edit , UpDown , Timer أما XPManifest فلا حاجة لنا به إذ وظيفته إعطاء شكل ويندوز أكس بي للنموذج, و نحن في هذا البرنامج لن نمكن المستخدم من رؤية هذه الواجهة.
و نضبط خصائص الأداة UpDown1 كما يلي: Associate = Edit1

و يتم هذا عن طريق تحديد الأداة UpDown1 ثـم الإنتقـال إلـى قائمـة الخـصائص و البحـث عـن الخاصـية Associate وضبطها على Edit1

Associate | Edit1

و نضبط خصائص الأداة edit1 كما يلي: edit1.text = 0
في الحقيقة بمجرد ضبط الخاصية Associate = Edit1 للأداة updown1 فإن خاصية edit1.text = 0 سوف تحدث آليا.
الآن نقوم بحفظ البرنامج في مجلد نسـميه islam نقوم بإنشائه يدويا فـي القـرص الـصلب c و نـضع داخل هذا المجلد مجلدا فرعيا و نسميه test . و نضع في هذا المجلد مجموعة من الصور ذات نمط jpg و نرقمها من 0 إلى ما نشاء (أنا وضعت 9 صور كل صورة تحمل حديثا نبويا) كما يلي:



ومهام البرنامج هو فتح صورة من الصور في كل مرة يشـتغل فيها البرنامج و ليتم ذلك قم بالنقر المـزدوج على النموذج و اكتب الأمر التالي:

```
procedure TForm1.FormCreate(Sender: TObject);

begin
ShellExecute (Handle, 'open', PChar ('c:\islam\test\'+ edit1.text +'.jpg') ,  nil, nil, SW_SHOW);
end;
```

كي يعمل هذا الكود قم بإضافة الفئة ShellAPI إلى uses

```
interface

uses
  Windows,Registry, Messages, SysUtils,
  Dialogs,mmsystem,CPL,ShellAPI,  StdCtı

type
```

حيث يقوم هذا الأمر بفتح ملف مـا مـن المـسار التـالي **c:\islam\test** أمـا اسـم الملـف فهنـا تكمـل المشـكلة فلو حددناه ب 0.jpg فسـيكون المسار على هذا النحـو **('c:\islam\test\0.jpg)** ممـا يعنـي أننا سـنكتب 10 أوامر (من صفر إلى 9) و إذا كان عدد الصور (الأحاديث) التي أرغب في عرضها 300 فهـذا يعني أنني مجبر على كتابة 301 كود مع تغيير إسم الملف **0.jpg 1.jpg 2.jpg ..... 300.jpg**. لتجنـب هذه المشـقة رأيت أن أجعـل إسـم الملـف هـو محتـوى (نـص) الأداة edit1 + الـنص **'.jpg'** فـإذا قمنـا بتشغيل البرنامج سيقوم بالذهاب إلى المجلد test و البحث عن الملف الذي يحمل الإسم التالي:
إذا كان محتوى الأداة edit1 يساوي 4 فإنه سيتم إضافتها إلى النص **'.jpg'** فيصبح اسـم الـصورة التـي سيظهرها البرنامج من المسار **'c:\islam\test\'** هي **4.jpg** فيصبح و كأننا كتبنا الأمر التالي

```
ShellExecute (Handle, 'open', PChar ('c:\islam\test\4.jpg') ,  nil, nil, SW_SHOW);
```

لابد و أنك أدركت أننا أردنا جعل محتوى النص متغييرا آليا فيكون الصفر عند أول إستخدام ثم ينتقل للرقم 1 .....إلى غاية بلوغ العدد النهائي وفقا لعدد الصور التي وضعتها فإذا كنت قد وضعت تسع صور فلابد لـك من إنشاء أمر يجعل محتوى النص يعود إلـى الـصفر (يستحـسن وضـع مـا يزيـد عـن 100 صـورة حتـى لا يشعر المستخدم بأنه يقرأ نفس الأحاديث).

ما نريده الآن هو كيفية تغيير محتوى النص آليا. يكو ن هذا وفق الكود التالي:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
ShellExecute (Handle, 'open', PChar ('c:\islam\test\'+ edit1.text +'.jpg') ,  nil, nil, SW_SHOW);
updown1.Position:=StrToint(Edit1.Text)+1 ;
end;
```

الكود الذي أضفناه يقوم بجعل وضع الأداة updown1 = محتوى الأداة edit1+1 أما الدالة StrToint فتمكننا من تحويل محتوى الأداة النصية إلى حالة رقمية. و مجمل القول هو أن البرنامج عند اشتغاله سيقوم بتنفيذ مايلي:

```
ShellExecute (Handle, 'open', PChar ('c:\islam\test\0.jpg') ,  nil, nil, SW_SHOW);
```

ثم ينظر إلى محتوى الأداة Edit1 فإذا و جده = 0 فإنه يضيف له الرقم 1 فيصبح 1 فتصبح الصورة (الحديث) التي ستفتح المرة القادمة هي

```
ShellExecute (Handle, 'open', PChar ('c:\islam\test\1.jpg') ,  nil, nil, SW_SHOW);
```

ثم ينظر إلى محتوى الأداة Edit1 مجددا فإذا و جده = 1 فإنه يضيف له الرقم 1 فيصبح 2 فتصبح الصورة (الحديث) التي ستفتح المرة القادمة هي

```
ShellExecute (Handle, 'open', PChar ('c:\islam\test\2.jpg') ,  nil, nil, SW_SHOW);
```

فإذا بلغ عدد الصور تسعة سيقوم البرنامج بالعودة إلى الصورة رقم 0 و ذلك وفقا للكود التالي

```
procedure TForm1.FormCreate(Sender: TObject);
begin
ShellExecute (Handle, 'open', PChar ('c:\islam\test\'+ edit1.text +'.jpg') ,  nil, nil, SW_SHOW);
updown1.Position:=StrToint(Edit1.Text)+1 ;
if  strToint(edit1.text) =9 then
 Updown1.position:= 0 ;
end;
```

لو قمت بتنفيذ البرنامج ثم إغلاقه فتشغيله من جديد فإنك تلاحظ أن Edit1.Text=1 و هذا لأن البرنامج لا يحتفظ بآخر حالة كان عليها بل إنه يعود لحالة التصميم و هو Edit1.Text=0 فإذا أضيف له الرقم 1 و فقا للكود السالف الذكر فسيكون المحتوى هو 1 و سيحدث هذا في كل مرة. أما ما يجب فعله فهو أن يحتفظ البرنامج بآخر حالة عليه. فإذا كانت العملية السالفة قد تمت بنجاح و أن Edit1.Text= 1 فإن الحالة هذه يجب ألا تتغير حتى تصبح العملية القادمة هي 1+1 = 2 وبالتالي تشغيل الصورة التي تحمل هذا الإسم. و كي يحتفظ البرنامج بآخر حالة كان عليها ثمة دالة سنضيفها للبرنامج في آخر عملية البرمجة.

الآن وجب علينا إنشاء الأمر الذي يجعل البرنامج يعمل مع بدء التشغيل.

```
procedure TForm1.FormCreate(Sender: TObject);
var
  reg: TRegistry;
begin
ShellExecute (Handle, 'open', PChar ('c:\islam\test\'+ edit1.text +'.jpg') ,  nil, nil, SW_SHOW);
updown1.Position:=StrToint(Edit1.Text)+1 ;
if  strToint(edit1.text) =9 then
 Updown1.position:= 0 ;
reg := TRegistry.Create;
  reg.RootKey := HKEY_LOCAL_MACHINE;
  reg.LazyWrite := false;
  reg.OpenKey('Software\Microsoft\Windows\CurrentVersion\Run',
            false);
  reg.WriteString('My App', Application.ExeName);
  reg.CloseKey;
  reg.free;
end;
```

أما الآن نقوم بالنقر المزدوج على الأداة Timer1 و كتابة الكود التالي و الذي يقوم بإغلاق البرنامج بعد لحضات من تشغيلة. و بالتالي فإن ما يشاهده المستخدم هو الصورة التي تحوي الحديث النبوي و ليس البرنامج الذي صممناه.

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
form1.close;
end;
```

يمكنك حفظ البرنامج ثم إعادة تشغيل الويندوز و ستظهر لك الصورة رقم 1 و ستظل نفس الصورة تظهر مع كل تشغيل مالم تضف الكود التالي:

1 - قم بإضافة مايلي

```
public
    constructor Create(AOwner: TComponent); override; // (1)
  procedure BeforeDestruction; override;  // (2)
end;
```

- 2

```
var
  Form1: TForm1;
  PreservePath: string; // (3)
implementation
```

-3

```
{$R *.dfm}

constructor TForm1.Create(AOwner: TComponent); // (4)
begin

  PreservePath := ExtractFilePath(Application.ExeName) +
    'Preserve';
  if not DirectoryExists(PreservePath) then
    CreateDir(PreservePath);
  PreservePath := PreservePath + '\';

  if FileExists(PreservePath + ClassName + '.sav') then
  begin
    CreateNew(AOwner, 0);
    with TFileStream.Create(PreservePath + ClassName + '.sav',
                            fmOpenRead or fmShareDenyWrite) do
    try
      ReadComponent(Self);
    finally
      Free;
    end;
  end
  else

  inherited Create(AOwner);

end;
```

-4

```
procedure TForm1.BeforeDestruction; // (5)
begin
  inherited;

  with TFileStream.Create(PreservePath + ClassName + '.sav',
    fmCreate) do
```

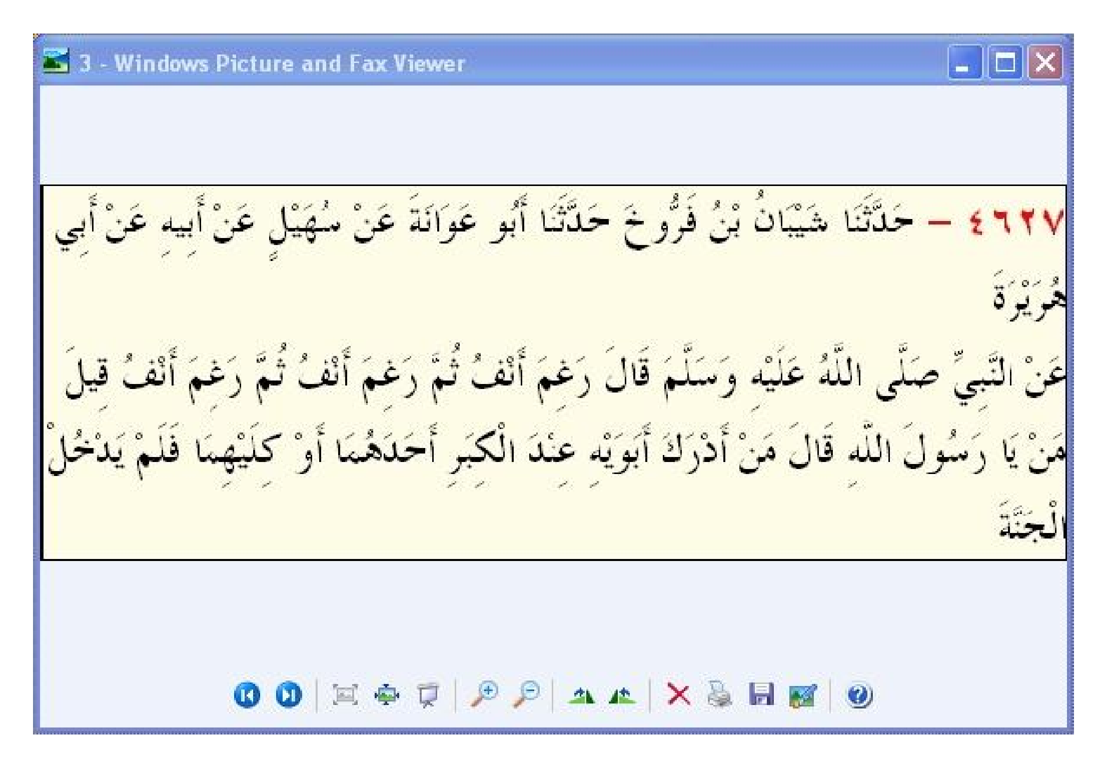
```
  try
    WriteComponent(Self);
  finally
    Free;
  end;
end;


end.
```

ملاحظة تم الإستعانة بهذه الدالة من موقع خالد شقروني.

http://www.shagrouni.com/arabic/index.html

بهذا نكون قد أنهينا برنامجنا و لا نحتاج لشيء سوى تحزيمه على شرط استخراج البرنامج و المجلد الفرعي TEST الذي يضم الصور في هذا المسار c:\islam و هذا العمل لا يحتاج إلى عبقرية خصوصا مع توفر عدد كبير من البرامج الخاصة بعمل SETUP .



هذا نموذج لما يظهر مع بعد التشغيل. يمكنك التلاعب بهيئة الملفات كما تشاء كما يمكنك إظهار واجهة البرنامج و تمكين المستخدم من ضبط بعض الخصائص كأن يختار نوع الأحاديث وفقا للموضوع أو اختيار ملفات صوتية أو ملفات الفلاش بلاير التي تكون قد صممتها أو حصلت عليها من عند زميل مختص في هذا المجال كي تبني برامج محترمة .

عموما, يمكنك بناء عدة برامج وفقا للطريقة السابقة.

في الأخير أنوه أن مشاريع أخرى كنت بصدد اعدادها , غير أن انصرافي عن مجال المبرمجة قد يطول فأحببت أن أختم الكتيب عند هذا الحد عسى أن ينتفع القراء بما يحويه من معلومات.

كما أحيلكم إلى هذه المصادر الهامة لتعلم لغة دلفي
موقع المبرمج الإيطالي ماركو كانتو
www.marcocantu.com

يمكنكم تحميل كتابه أساسيات باسكال مجانا و باللغة العربية

موقع الفريق العربي للبرمجة, و هو أحد أغنى المواقع بالمادة العلمية الراقية المتوفرة على صفحات منتدياته المختلفة.
www.arabteam2000.com
لقد تم الإستعانة ببعض المواضيع الموجودة على صفحات هذا الموقع في كتابنا هذا.

إعداد : **يوسف بوسعيد**
You_per7@hotmail.com